The Vertex Cover Problem

- Given an undirected graph $G = (V, E)$, a vertex cover of G is a subset U of V such that for every edge e in E, at least one endpoint of E belongs to U
- The vertex cover problem asks us to find a minimum-cardinality vertex cover
- Given a graph G and a bound B, the decision version of the vertex cover problem, denoted VC, asks us whether G has a vertex cover of size at most B
- It is easy to argue that VC belongs to NP
- We will use a reduction from IS (independent set) to prove that VC is NP-complete

IS $\leq_P$ VC (and VC $\leq_P$ IS)

- Claim: For any graph $G = (V, E)$, U is a vertex cover of G if and only if $V \setminus U$ is an independent set of G
  - If U is a vertex cover of G, then no edge has both endpoints in $V \setminus U$, and hence $V \setminus U$ is an independent set of G
  - If $V \setminus U$ is an independent set of G, then every edge has at least one endpoint in U, and hence U is a vertex cover of G
- Thus, a graph $G = (V, E)$ has an independent set of size at least B if and only if G has a vertex cover of size at most $|V| - B$

A Greedy Algorithm for Vertex Cover

- Let $G = (V, E)$ be a given graph
- Initialize U to the empty set
- While U is not a vertex cover
  - Let (u, v) be an edge in E such that $U \cap \{u, v\} = \emptyset$
  - Add u and v to U
- Return U

Analysis of the Greedy Algorithm

- In each iteration, we grow U by selecting an uncovered edge (u, v) and adding its endpoints to U
- Let E' denote the set of selected edges
- The vertex cover output by the algorithm is of size $2|E'|$
- The $2|E'|$ endpoints of the edges in E' are all distinct
- Thus any vertex cover has size at least $|E'|$

Approximability of IS and VC

- Earlier we established the NP-completeness of VC by showing that it is essentially equivalent to IS
- Notwithstanding this result, our 2-approximate algorithm for vertex cover does not imply a 2-approximate algorithm for independent set
  - Consider a graph $G = (V, E)$ for which the size of a maximum independent set is k where $1 \leq k \leq |V|/2$
  - Thus the size of a minimum vertex cover is $|V| - k \geq |V|/2$, and so V is a 2-approximate solution to the vertex cover problem
  - On the other hand, the complement of V (i.e., the empty set) is not an α-approximate solution to the maximum independent set problem for any α

Approximation-Preserving Reductions

- Reductions play a central role in the theory of approximability, just as they do in the theory of NP-completeness
- In the approximability setting, we seek reductions that are "approximation preserving"
  - Given a polynomial-time α-approximation algorithm for one problem, we show how to obtain a polynomial-time β-approximation for another problem
  - Typically $\beta \geq \alpha$ and we strive to keep the ratio β/α small
  - Sometimes we can improve the approximation guarantee (i.e., $1 < \beta < \alpha$)
- As discussed on the previous slide, the standard reduction from IS to VC is not approximation preserving

Inapproximability of Vertex Cover

- Dinur and Safra (2005) proved that if P $\neq$ NP then there is no 1.3606-approximate polynomial-time algorithm for the vertex cover problem
- Khot and Reghev (2008) proved that if the unique games conjecture holds then for any constant $\varepsilon > 0$ there is no $(2 - \varepsilon)$-approximate polynomial-time algorithm for the vertex cover problem
  - It is known that if the unique games conjecture holds then P $\neq$ NP
- The proof of such results is beyond the scope of the present course

The Weighted Vertex Cover Problem

- Let $G = (V, E)$ be a vertex-weighted graph
  - Each vertex v in V has a nonnegative weight w_v
- The weight of a vertex cover U is defined as

$$w(U) = \sum_{v \in U} w_v$$

- We will present two 2-approximate algorithms for the weighted vertex cover problem

A 2-Approximation Based on LP Rounding

- It is easy to formulate the weighted vertex cover problem as a 0-1 integer linear program (ILP)
  - There is a 0-1 variable x_v for each vertex v in V
  - There is a constraint $x_u + x_v \geq 1$ for each edge (u, v) in E
  - The objective is to minimize $\sum_{v \in V} w_v x_v$
- The first step towards developing our approximation algorithm is to "relax" the 0-1 constraints on the x_v's

LP Relaxation for the Weighted Vertex Cover Problem

- Relaxing the 0-1 constraints on the x_v's, we obtain the following linear program
  - There is a nonnegative variable x_v for each vertex v in V
  - There is a constraint $x_u + x_v \geq 1$ for each edge (u, v) in E
  - The objective is to minimize $\sum_{v \in V} w_v x_v$
- The above linear program can be solved in polynomial time
- Let x^* be an optimal solution
- The next step in our approximation algorithm is to "round" the fractional solution x^* to a 0-1 solution

The Rounding Step

- For each vertex v in V, we set x_v to 1 if $x_v^* \geq \frac{1}{2}$, and to 0 otherwise
- The rounded solution x is feasible
  - For any edge (u, v) in E, we have $x_u^* + x_v^* \geq 1$
  - Thus $\max(x_u^*, x_v^*) \geq \frac{1}{2}$
  - Thus either $x_u = 1$ or $x_v = 1$
- The objective function value for x is at most twice that for x^*
  - For any vertex v in V, we have $x_v \leq 2x_v^*$
  - Thus $\sum_{v \in V} w_v x_v \leq 2 \sum_{v \in V} w_v x_v^*$

General Comments on LP Rounding

- LP rounding is a powerful technique that yields good approximation guarantees for many optimization problems
- Sometimes we employ a more sophisticated rounding method than that used for weighted vertex cover
- Sometimes LP rounding does not seem to work
  - For example, all natural rounding schemes may lead to infeasible solutions or to poor approximation guarantees
- A drawback of the LP rounding technique is that it requires us to solve an LP
  - While linear programming has polynomial time complexity, it may be too slow for extremely large problem instances

Weighted Vertex Cover: A Faster 2-Approximate Algorithm

- We now present a faster 2-approximate algorithm for weighted vertex cover
  - The algorithm is based on maintaining a "price" p_e for each edge e in E
  - The price of any edge is initially zero, and never decreases
  - We will maintain the invariant that for every vertex v in V, $\sum_{(u,v)\in E} p_{(u,v)} \leq w_v$
  - We say that a vertex v is "tight" if $\sum_{(u,v)\in E} p_{(u,v)} = w_v$
  - Since edge prices never decrease, once a vertex becomes tight, it remains tight
  - The algorithm selects the tight vertices for the vertex cover
  - The algorithm terminates as soon as the tight vertices form a vertex cover

The Algorithm

- Initialize p_e to 0 for all e in E
- While the set of tight vertices is not a vertex cover
  - Let (u, v) be an edge in E such that neither u nor v is tight
  - Raise $p_{(u,v)}$ until either u or v becomes tight
- Output the set of tight vertices

Analysis

- It is easy to see that the algorithm outputs a vertex cover
- Moreover, the invariant $\sum_{(u,v)\in E} p_{(u,v)} \leq w_v$ is maintained throughout for all vertices v
- For any edge e in E, let p^*_e denote the final price of e
- Lemma 1: For any vertex cover U, we have $\sum_{e\in E} p^*_e \leq w(U)$
  - Summing our invariant (in the final state) over all v in U, we obtain $w(U) \geq \sum_{v\in U} \sum_{(u,v)\in E} p^*_{(u,v)}$
  - Since U is a vertex cover, the latter sum is at least $\sum_{e\in E} p^*_e$

Analysis cont'd

- Let U^* denote the set of all tight vertices in the final state
- Lemma 2: $w(U^*) \leq 2\sum_{e \in E} p_e^*$
    - One way to see this is to create a pile of money at each vertex v as follows: For each edge (u, v) in E, add $p_{(u,v)}^*$ dollars to the pile at v
    - The total amount of money distributed to the vertices is $2\sum_{e \in E} p_e^*$
    - The total amount of money distributed to the vertices in U^* is $w(U^*)$
    - Since the edge prices are nonnegative, the claimed inequality holds

Analysis (cont'd)

- We have established the following two lemmas, where U^* denotes the vertex cover returned by the algorithm
  - Lemma 1: For any vertex cover U, we have $\sum_{e \in E} p_e^* \leq w(U)$
  - Lemma 2: $w(U^*) \leq 2 \sum_{e \in E} p_e^*$
- Combining these two bounds, we find that $w(U^*) \leq 2w(U)$ for any vertex cover U
- Thus the algorithm achieves the claimed approximation ratio of 2
- What methodology can we use to design such clever price-based approximation algorithms?
  - Next we give a primal-dual interpretation of the foregoing algorithm

The Dual of the Relaxed LP

- Recall the LP relaxation for the weighted vertex problem
  - There is a nonnegative variable x_v for each vertex v in V
  - There is a constraint $x_u + x_v \geq 1$ for each edge (u, v) in E
  - The objective is to minimize $\sum_{v \in V} w_x x_v$
- We can mechanically form the dual of the above primal LP
  - There is a nonnegative variable y for each edge e in E
  - There is a constraint $\sum_{(u,v) \in E} y_{(u,v)} \leq w_v$ for each vertex v in V
  - The objective is to minimize $\sum_{e \in E} y_e$

A Primal-Dual Interpretation

- The edge prices in our 2-approximation algorithm correspond to the dual variables
  - Both are nonnegative
  - The dual constraint $\sum_{(u,v)\in E} y_{(u,v)} \leq w_v$ corresponds to the key invariant maintained with respect to the prices
- Can we use LP duality to guide the design and analysis of our 2-approximation algorithm?

Revisiting Lemma 1

- Lemma 1: For any vertex cover U, we have $\sum_{e \in E} p_e^* \leq w(U)$
- The quantity $w(U)$ corresponds to the value of the primal objective for the feasible 0-1 solution x corresponding to U
  - For any vertex v in V, we have $x_v = 1$ if v belongs to U, and $x_v = 0$ otherwise
- The LHS corresponds to the value of the dual objective for the solution y corresponding to p^*
  - For any edge e in E, we have $y_e = p_e^*$
  - This solution y is feasible for the dual because p^* satisfies the key invariant
- Thus Lemma 1 follows by weak duality

Revisiting the Design of the Algorithm

- We will iteratively update a 0-1 vector x (with a component for each vertex) and a nonnegative vector y (with a component for each edge)
  - Initially, y will be the all-zeros vector, which is feasible for the dual
  - Whenever we update y, we will do so by increasing some component while maintaining feasibility
  - We will use our updates of y to guide our updates of x
  - We will terminate once we arrive at a feasible x

A Connection to Complementary Slackness

- We cannot expect to obtain a feasible 0-1 primal solution x and a feasible dual solution y satisfying the complementary slackness conditions
  - If we did, it would imply that x is optimal, and we are only seeking an approximately optimal solution
  - It would also imply that the "integrality gap" of the LP is 1, meaning that the optimal objective function value of the relaxed LP is equal to that of the original 0-1 ILP
    - In fact, as we will show a bit later, the integrality gap is at least $2(1 - \frac{1}{|V|})$, which tends to 2 as $|V| \to \infty$
- Still, we can sometimes use a variation of the complementary slackness conditions to guide the design of our iterative updates

A Connection to Complementary Slackness

- One of the complementary slackness conditions states that if the dual constraint corresponding to a variable x_v is not tight, then x_v is 0
- This condition inspires us to use our current feasible dual solution y to determine a 0-1 primal solution x by setting x_v to 0 if and only if the dual constraint corresponding to x_v is not tight
  - This corresponds precisely to the method used in the price-based greedy algorithm to interpret the current prices as a set of selected vertices

Updating the Dual Variables

- How should we update the dual variables in each iteration?
- The reason we haven't terminated is that the 0-1 primal solution corresponding to y is infeasible, i.e., one or more primal constraints are violated
- We seek a simple way to update the dual-feasible solution y to a new dual-feasible solution y' such that the corresponding 0-1 solution x' violates fewer primal constraints than x
  - Suppose a primal constraint $x_u + x_v \geq 1$ is violated
  - It is natural to increase the corresponding dual variable $y_{(u,v)}$ to eliminate this violation, which is what the algorithm does
  - Rather than raising a single such violation-related dual variable $y_{(u,v)}$, we could raise all of them uniformly until some dual constraint becomes tight

The Integrality Gap of the Vertex Cover LP

- Let G be a complete graph on n vertices, where each vertex has weight 1
  - Remark: This instance corresponds corresponds to an unweighted instance
- A minimum vertex cover is of size at least $n - 1$
  - Correspondingly, the optimal objective function value for the 0-1 ILP corresponding to this weighted vertex cover instance is $n - 1$
- The relaxed LP admits a feasible fractional solution with objective function value $n/2$
  - We can set $x_v = \frac{1}{2}$ for all vertices v in V
- Hence the integrality gap is at least $\frac{n-1}{n/2} = 2(1 - \frac{1}{n})$

General Comments on Integrality Gap

- There are many NP-hard optimization problems for which the best approximability results known use LP-based techniques
  - LP rounding
  - Primal-dual methods
- The integrality gap is a significant barrier for such techniques